REFERENCES 83

[67] B. Randell, "System structure for software fault tolerance," SIGPLAN Not., vol. 10, p. 437–449, apr 1975.

- [68] N. Harrand, Software Diversity for Third-Party Dependencies. PhD thesis, KTH, Software and Computer systems, SCS, 2022. QCR 20220413.
- [69] J. V. Cleemput, B. Coppens, and B. De Sutter, "Compiler mitigations for time attacks on modern x86 processors," ACM Trans. Archit. Code Optim., vol. 8, jan 2012.
- [70] A. Homescu, S. Neisius, P. Larsen, S. Brunthaler, and M. Franz, "Profile-guided automated software diversity," in *Proceedings of the 2013 IEEE/ACM International Symposium on Code Generation and Optimization (CGO)*, pp. 1–11, IEEE, 2013.
- [71] S. Bhatkar, D. C. DuVarney, and R. Sekar, "Address obfuscation: an efficient approach to combat a board range of memory error exploits," in *Proceedings* of the USENIX Security Symposium, 2003.
- [72] S. Bhatkar, R. Sekar, and D. C. DuVarney, "Efficient techniques for comprehensive protection from memory error exploits," in *Proceedings of the* USENIX Security Symposium, pp. 271–286, 2005.
- [73] K. Pettis and R. C. Hansen, "Profile guided code positioning," in *Proceedings* of the ACM SIGPLAN 1990 Conference on Programming Language Design and Implementation, PLDI '90, (New York, NY, USA), p. 16–27, Association for Computing Machinery, 1990.
- [74] S. Crane, A. Homescu, S. Brunthaler, P. Larsen, and M. Franz, "Thwarting cache side-channel attacks through dynamic software diversity.," in NDSS, pp. 8–11, 2015.
- [75] A. Romano, D. Lehmann, M. Pradel, and W. Wang, "Wobfuscator: Obfuscating javascript malware via opportunistic translation to webassembly," in 2022 2022 IEEE Symposium on Security and Privacy (SP) (SP), (Los Alamitos, CA, USA), pp. 1101–1116, IEEE Computer Society, may 2022.
- [76] M. T. Aga and T. Austin, "Smokestack: thwarting dop attacks with runtime stack layout randomization," in *Proc. of CGO*, pp. 26–36, 2019.
- [77] S. Lee, H. Kang, J. Jang, and B. B. Kang, "Savior: Thwarting stack-based memory safety violations by randomizing stack layout," *IEEE Transactions on Dependable and Secure Computing*, 2021.
- [78] Y. Younan, D. Pozza, F. Piessens, and W. Joosen, "Extended protection against stack smashing attacks without performance loss," in 2006 22nd Annual Computer Security Applications Conference (ACSAC'06), pp. 429–438, 2006.

84 REFERENCES

[79] Y. Xu, Y. Solihin, and X. Shen, "Merr: Improving security of persistent memory objects via efficient memory exposure reduction and randomization," in *Proc. of ASPLOS*, pp. 987–1000, 2020.

- [80] G. S. Kc, A. D. Keromytis, and V. Prevelakis, "Countering code-injection attacks with instruction-set randomization," in *Proc. of CCS*, pp. 272–280, 2003.
- [81] E. G. Barrantes, D. H. Ackley, S. Forrest, T. S. Palmer, D. Stefanovic, and D. D. Zovi, "Randomized instruction set emulation to disrupt binary code injection attacks," in *Proc. CCS*, pp. 281–289, 2003.
- [82] M. Chew and D. Song, "Mitigating buffer overflows by operating system randomization," Tech. Rep. CS-02-197, Carnegie Mellon University, 2002.
- [83] D. Couroussé, T. Barry, B. Robisson, P. Jaillon, O. Potin, and J.-L. Lanet, "Runtime code polymorphism as a protection against side channel attacks," in IFIP International Conference on Information Security Theory and Practice, pp. 136–152, Springer, 2016.
- [84] S. Cao, N. He, Y. Guo, and H. Wang, "WASMixer: Binary Obfuscation for WebAssembly," arXiv e-prints, p. arXiv:2308.03123, Aug. 2023.
- [85] M. Jacob, M. H. Jakubowski, P. Naldurg, C. W. N. Saw, and R. Venkatesan, "The superdiversifier: Peephole individualization for software protection," in International Workshop on Security, pp. 100–120, Springer, 2008.
- [86] M. Henry, "Superoptimizer: a look at the smallest program," ACM SIGARCH Computer Architecture News, vol. 15, pp. 122–126, Nov 1987.
- [87] V. Le, M. Afshari, and Z. Su, "Compiler validation via equivalence modulo inputs," in *Proceedings of the 35th ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI '14, p. 216–226, 2014.
- [88] B. Churchill, O. Padon, R. Sharma, and A. Aiken, "Semantic program alignment for equivalence checking," in *Proceedings of the* 40th ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI 2019, (New York, NY, USA), p. 1027–1040, Association for Computing Machinery, 2019.
- [89] V. Le, M. Afshari, and Z. Su, "Compiler validation via equivalence modulo inputs," in *Proceedings of the 35th ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI '14, p. 216–226, 2014.